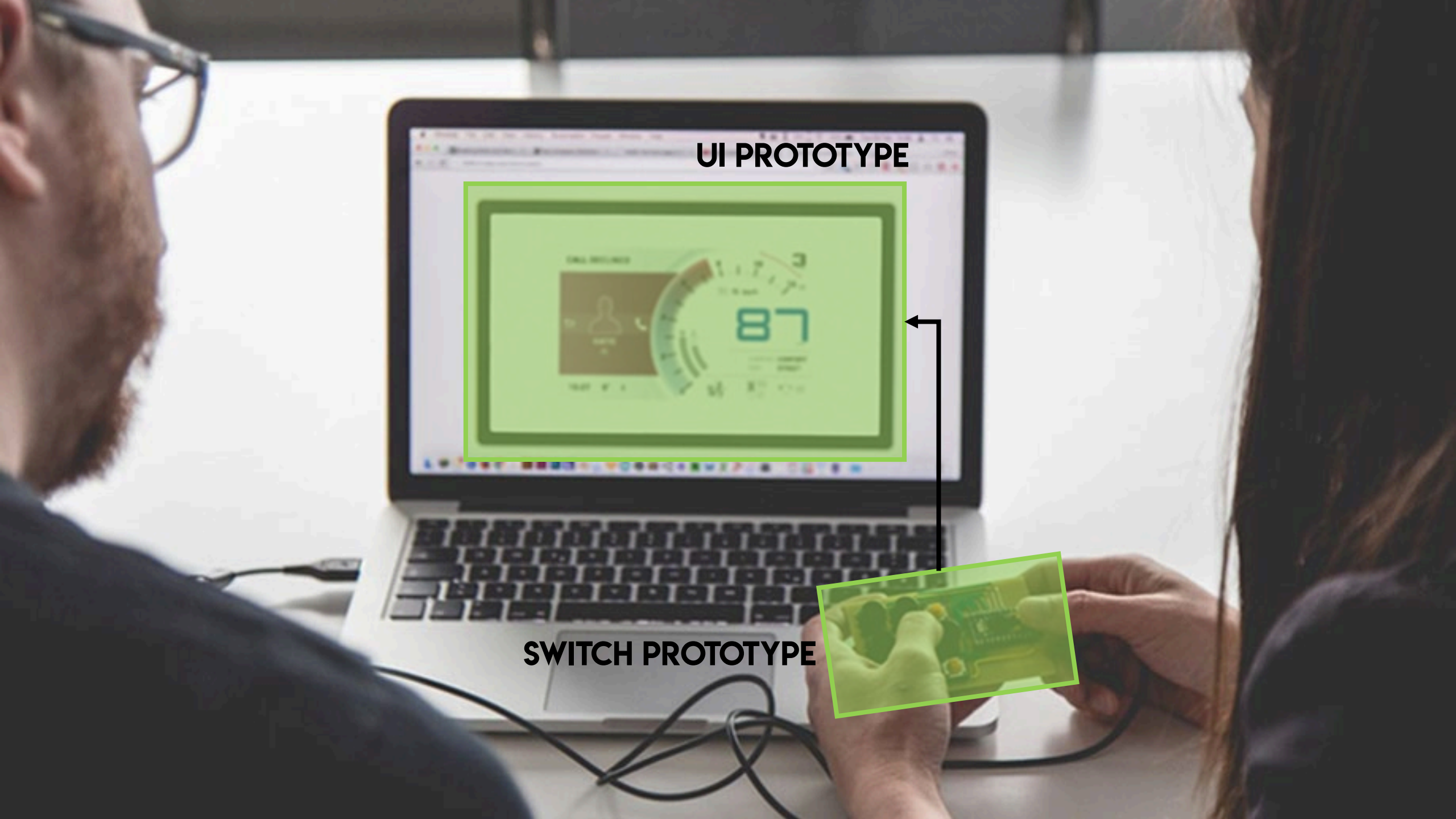
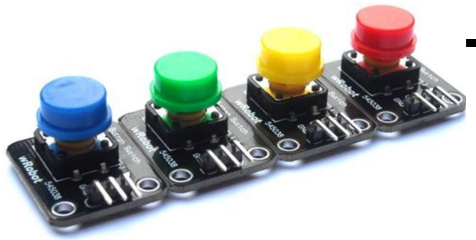
HMI PROTOTYPING


UI
SWITCHES
SWITCHES


UI PROTOTYPE
87
SWITCH PROTOTYPE

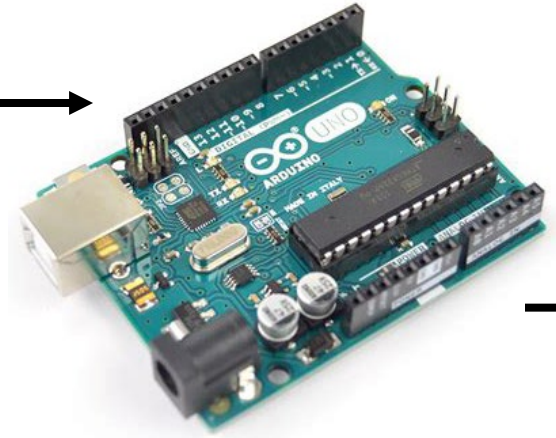

SET

ARDUINO AS A COMPUTER INPUT



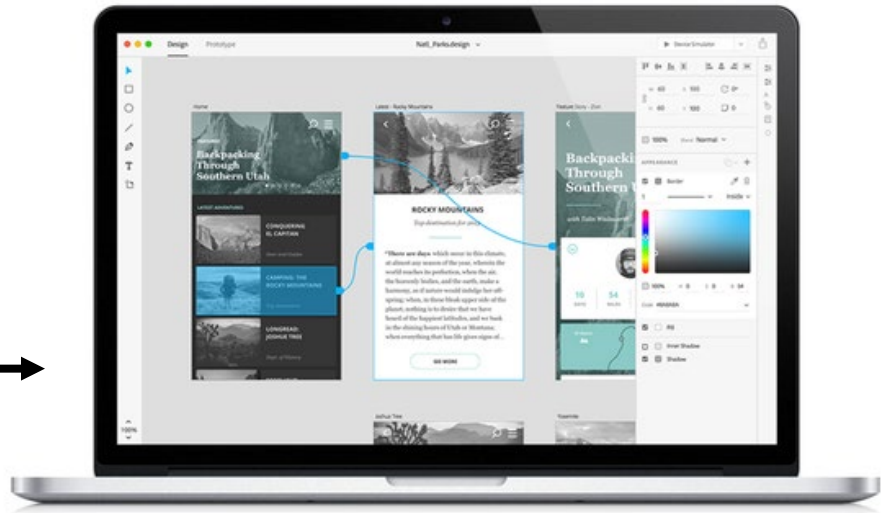
ANY SWITCH

(see notes at start of lecture)



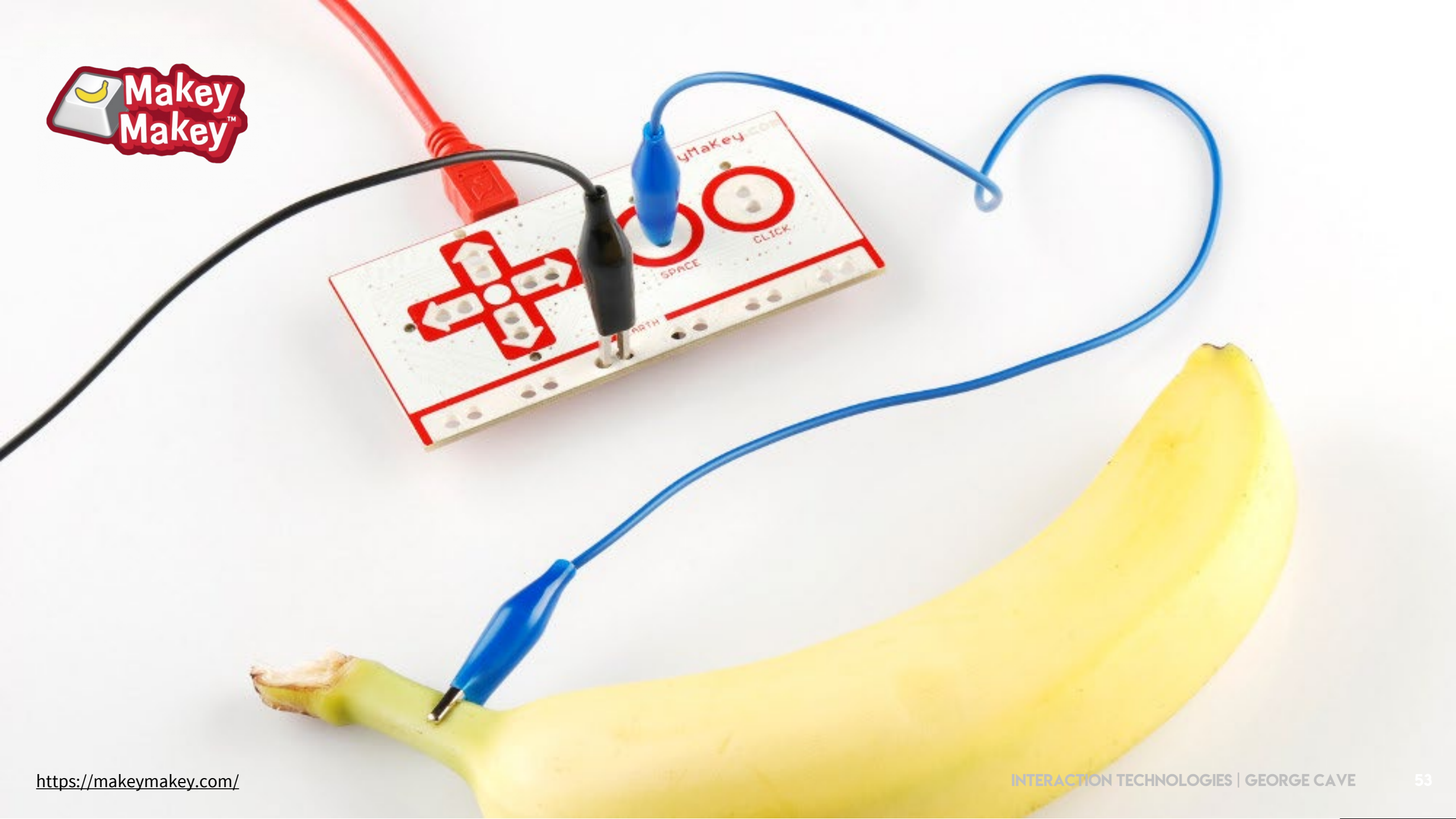
HID

```
Serial.println("left_button");
delay(500);
Serial.println("right_button");
delay(500);
```



LAPTOP

Any UI prototyping tool, e.g. Adobe Xd



https://makeymakey.com/

INTO TO ITSY BITSY 32U4 5V

Same as the Elegoo Uno R3, except…

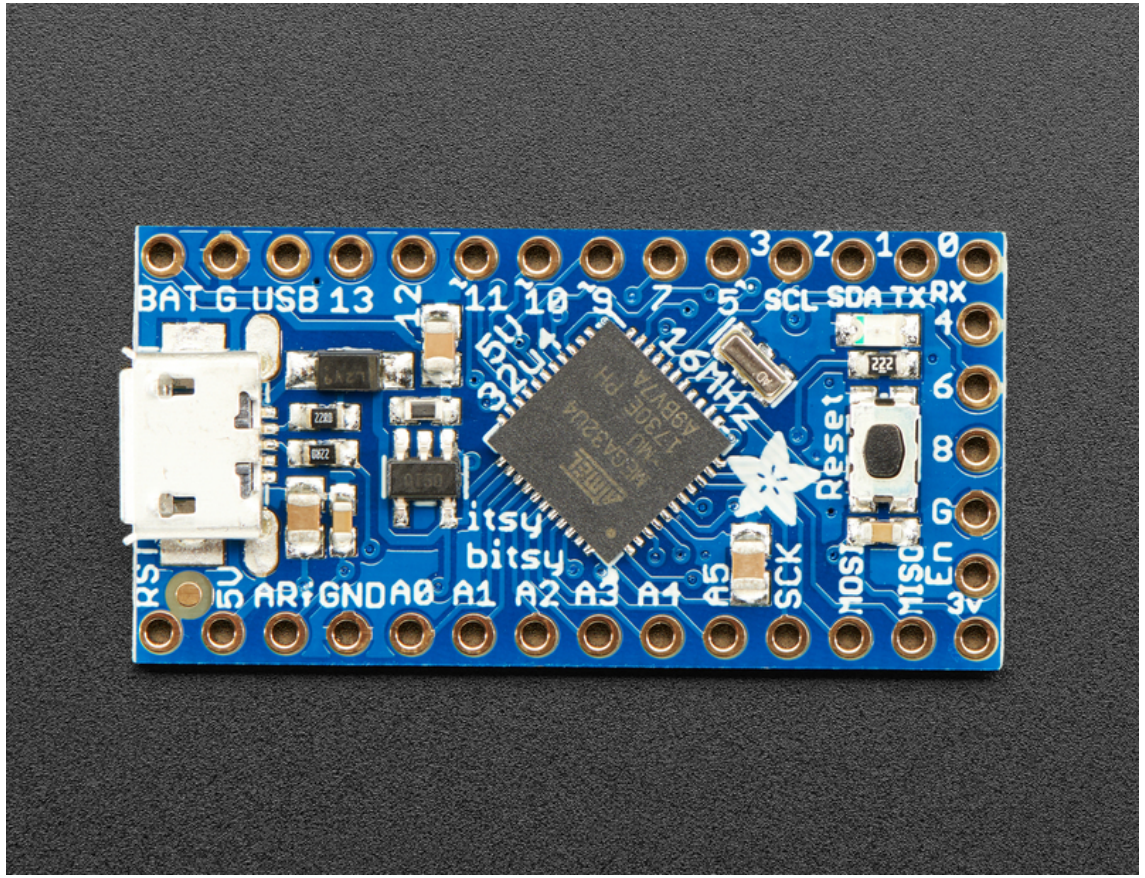
- Much smaller!
- Native USB -> can identify as a Keyboard, Mouse, Joystick etc…

Task:

1. Setup and test BLINK on ItsyBitsy 32u4 5V https://learn.adafruit.com/introducting-itsy-bitsy-32u4/arduino-ide-setup

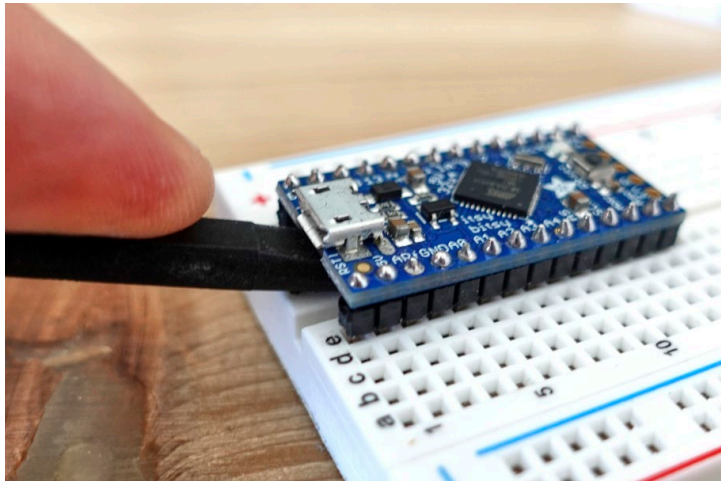
Notes:

- Take care when inserting into breadboard.
- Use a small plastic tool to remove, don't push all the way in, or connect male-female jumpers to legs instead.

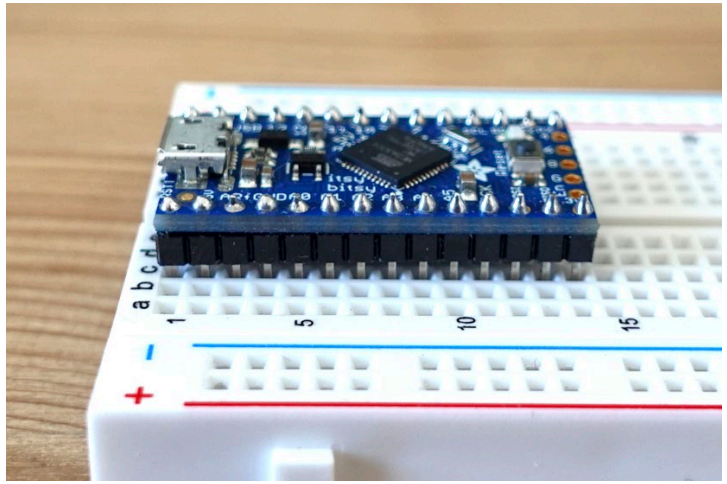


ITSY BITSY TIPS

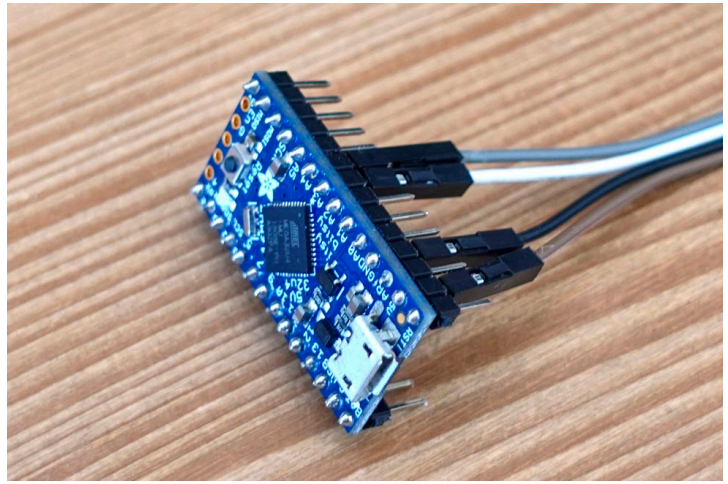
Take care when inserting into breadboard – it can be a very tight fit!



Use a small plastic tool to remove bit by bit from alternate ends



Or don’t push the Itsy Bitsy all the way into the breadboard to begin with.



Or use male->female jumper wires directly to the pins

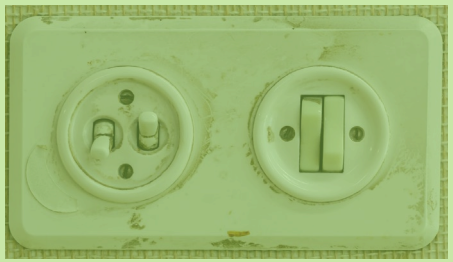
THREE BASIC INPUT TYPES

SWITCHES

MOMENTARY PUSH BUTTON

PUSH BUTTON

TOGGLE / ROCKER



SLIDE

MOMENTARY

LATCHING

DIALS

ABSOLUTE

INCREMENTAL

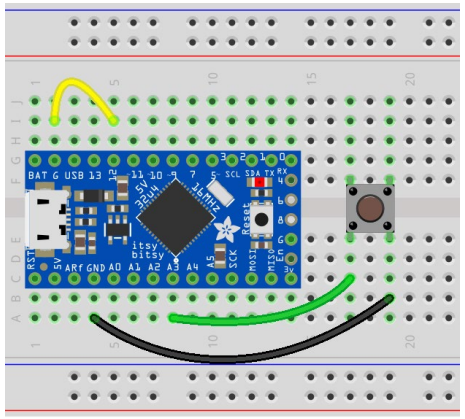
*CONTINUOUS

POTENTIOMETER

ENCODER

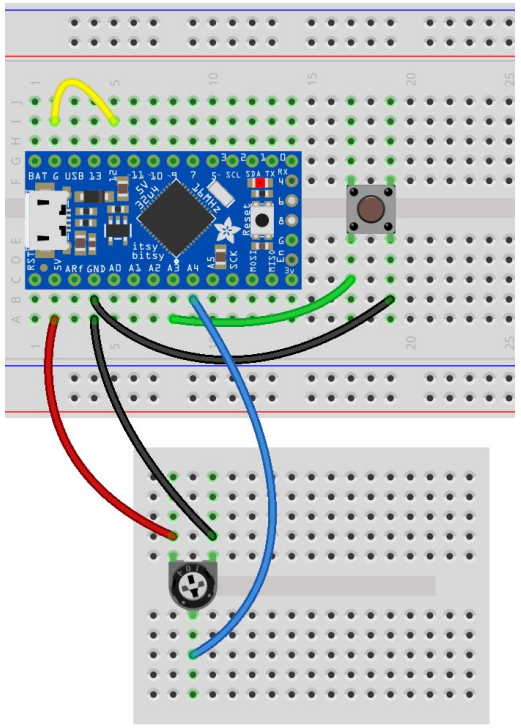
THREE BASIC INPUT TYPES

MOMENTARY PUSH BUTTON



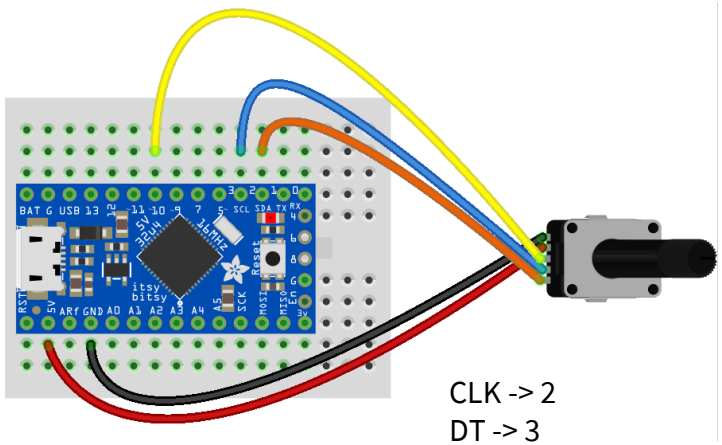
Push button will pull pin to GND when pressed

POTENTIOMETER

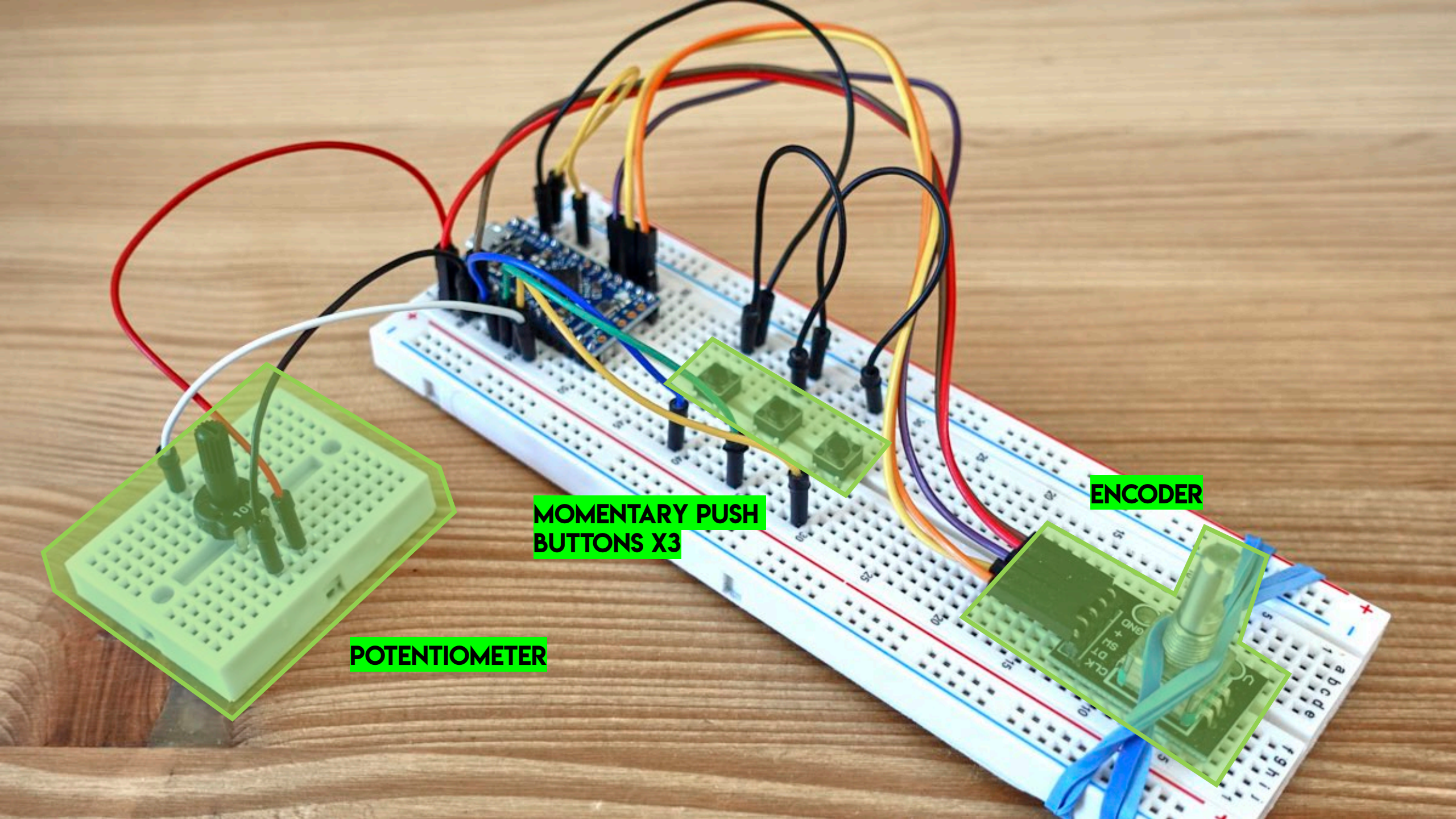


*Elegoo potentiometer makes better contact in small breadboard

ENCODER



CLK -> 2
DT -> 3
SW (push button) -> any pin
+ -> 5V
GND -> GND


MOMENTARY PUSH BUTTONS X3
ENCODER
POTENTIOMETER

KEYBOARD BUTTONS

Three basic parts:

1. `#include <Keyboard.h>`
2. `Keyboard.begin();`
3. `Keyboard.write('a');`

STOP!

This is dangerous – Arduino becomes a Keyboard continuously writing characters.

There might be no way back!

```
#include <Keyboard.h>

// Pin for button
const int button_pin = A3;

void setup(){

   // Set push button pin
   pinMode(button_pin, INPUT_PULLUP);

   // Start keyboard
   Keyboard.begin();

}

void loop(){
   // If button pushed, print character 'a'
   if(digitalRead(button_pin) == LOW){
      Keyboard.write('a');
   }
}
```

KEYBOARD BUTTONS

Add safety stop with pin 12.

Program will not begin until pin 12 connected to GND.

Code: [Arduino_sketches/01_Keyboard_Button]

Tasks:

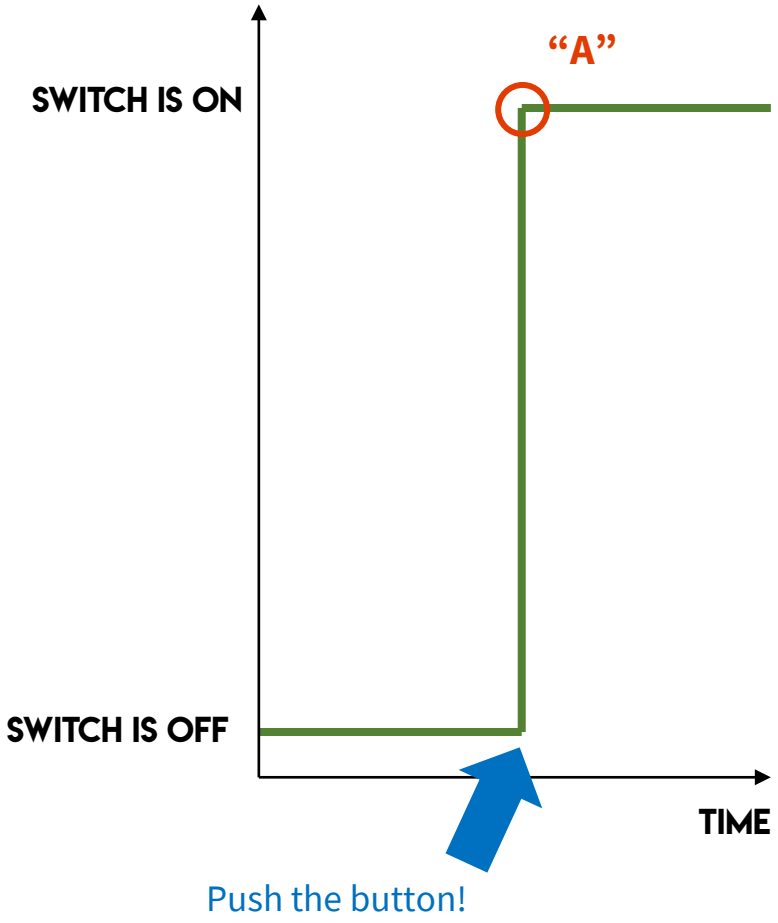
1. Modify to only send key once per button press
2. Add a second button with a different letter
3. Use buttons to send LEFT/RIGHT arrows rather than letters

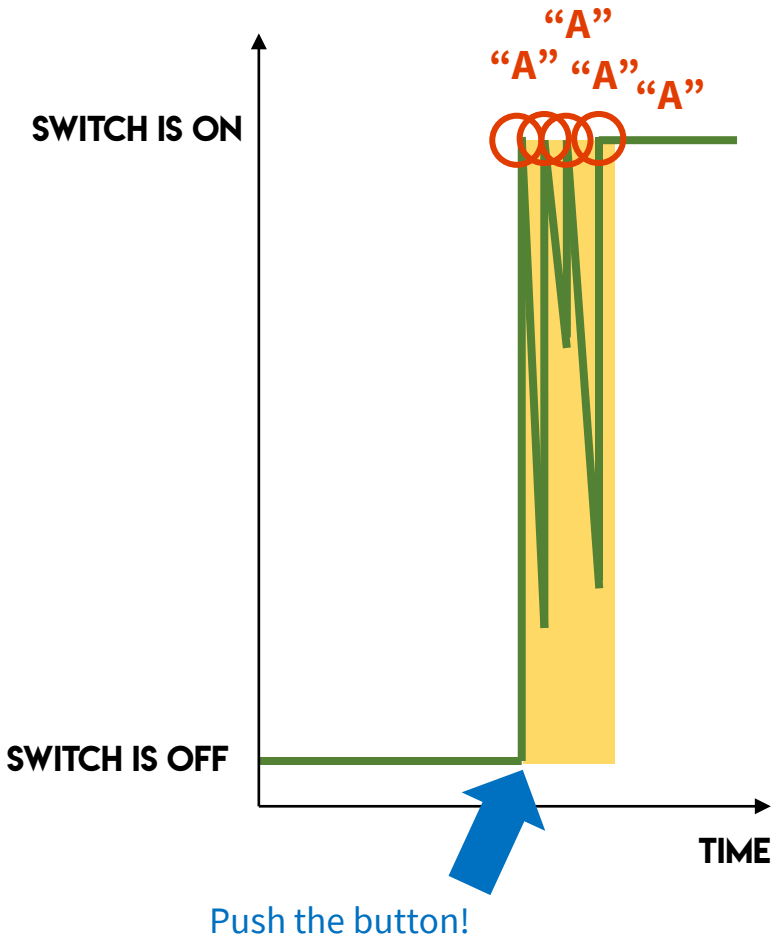
Hints:

- https://www.arduino.cc/reference/en/language/functions/usb/keyboard/
- https://www.arduino.cc/en/Reference/KeyboardModifiers

```
#include <Keyboard.h>

// Pin for button
const int button_pin = A3;

void setup(){

   // ⚠ Safety stop!
   pinMode(12, INPUT_PULLUP);
   while(digitalRead(12)==HIGH){
      // do nothing!
      delay(500);
   }

   // Set push button pin
   pinMode(button_pin, INPUT_PULLUP);

   // Start keyboard
   Keyboard.begin();

}

void loop(){
   // If button pushed, print character 'a'
   if(digitalRead(button_pin) == LOW){
      Keyboard.write('a');
   }
}
```

DE-BOUNCING


SWITCH IS ON
SWITCH IS OFF
TIME
"A"
Push the button!


SWITCH IS ON
SWITCH IS OFF
TIME
"A"
"A" "A" "A"
Push the button!

DE-BOUNCING

One way to solve using software is using the Bounce2 library by Thomas O Frederick.

Bounce2 includes lots of helpful functions to detect button states and changes:

https://thomasfredericks.github.io/Bounce2/files/class_bounce.html

Hints:

- Make sure button1.update() is called as frequently as possible (no delay() in loop()).
- https://github.com/thomasfredericks/Bounce2

Include Bounce2 library

Create a new debouncer called **button1**

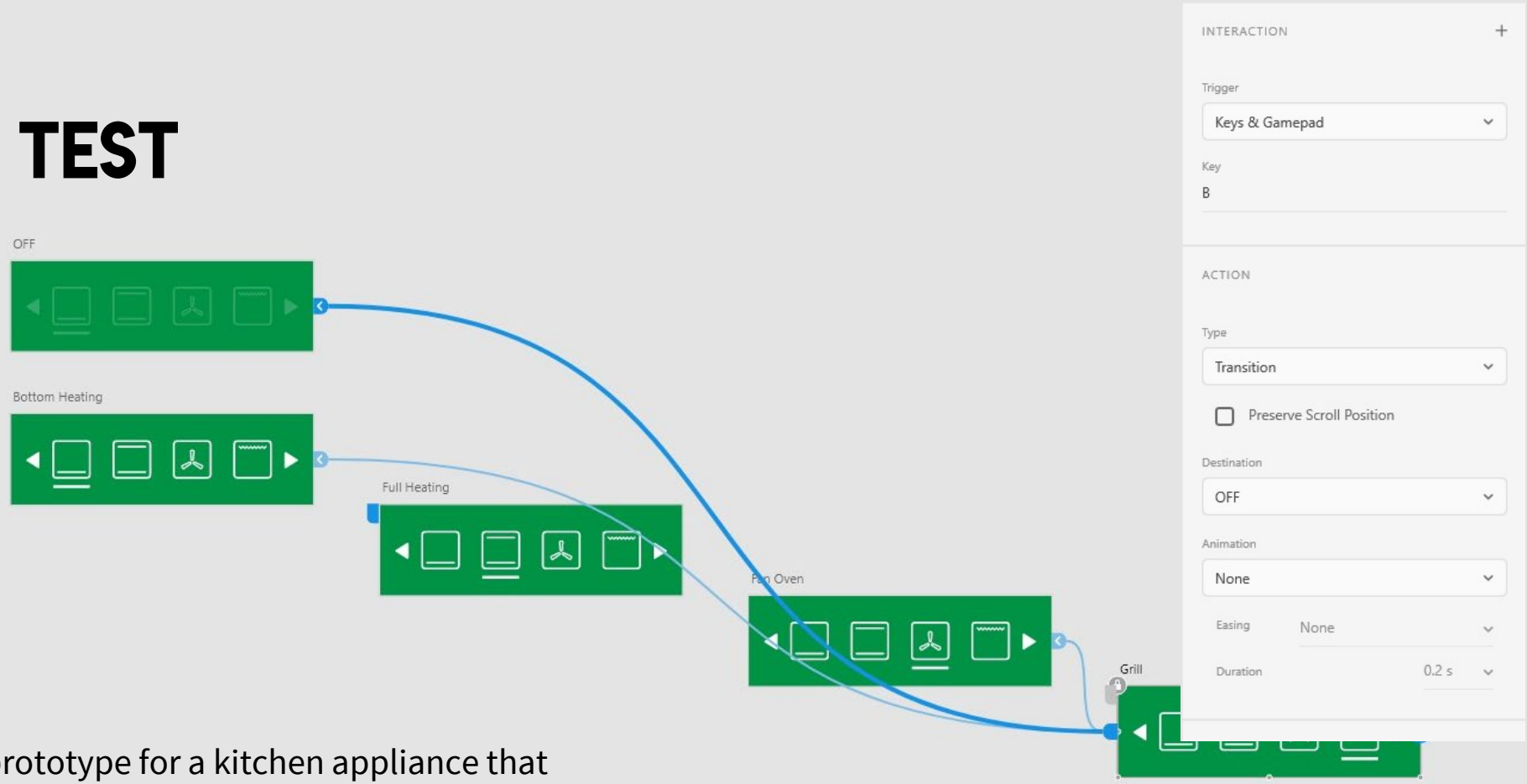
Attach the pin to the **button1**

Update the **button1** state with ever loop

If the button rose (pushed->released) then …

```
// Include the Keyboard and Bounce2 libraries
#include <Keyboard.h>
#include <Bounce2.h>

// Debouncers for buttons
Bounce button1 = Bounce();

void setup(){

   // ⚠ Safety stop!
   // Program will not begin unless pin 12 connected to GND
   pinMode(12, INPUT_PULLUP);
   while(digitalRead(12)==HIGH){
      // do nothing!
      delay(500);
   }

   // Attach pins A3 and A4 to debouncers
   button1.attach(A3,INPUT_PULLUP);

   // Start keyboard
   Keyboard.begin();
}

void loop(){

   // Update debouncers
   button1.update();

   // If buttons were released
   if(button1.rose()) {
      Keyboard.write(KEY_LEFT_ARROW);
   }
}
```

ADOBE XD UI TEST



Example prototype: [Adobe XD]

Task:

1. Create a simple interface prototype for a kitchen appliance that uses keyboard keys to navigate the interface.

Hints:

- Use letter keys not arrow keys for triggers (arrow keys automatically advance the artboards in Xd)
- https://helpx.adobe.com/xd/help/keys-gamepad-triggers.html

POTENTIOMETER

Basic code for potentiometer:

[Arduino_sketches/02_Keyboard_Potentiometer]

Tasks:

1. Print a single number from 0->8 rather than 0->1023 (much easier to control UI proto if Arduino only sends a single character)
2. Do not send character unless the value changes from last time
3. Add a small tolerance, so that output does not “flicker” between two adjacent values (e.g: if dial rests on boundary between two numbers)

Hints:

- Use map() to rescale value: https://www.arduino.cc/reference/en/language/functions/math/map/

Potentiometer pin

Read current value

If value changed, print to Keyboard

```
#include <Keyboard.h>

// Pin for button
const int pot_pin = A5;

void setup(){

   // ⚠ Safety stop!
   // Program will not begin unless pin 12 connected to GND
   pinMode(12, INPUT_PULLUP);
   while(digitalRead(12)==HIGH){
      // do nothing!
      delay(500);
   }

   // Start keyboard
   Keyboard.begin();
}

int last_val = 0;

void loop(){

   int current_val = analogRead(pot_pin);

   // If the value is different to the last printed value
   if(last_val != current_val){
      Keyboard.println(current_val);
      last_val = current_val;
   }
}
```

ENCODER

A very good library for reading Encoder changes is "Encoder" by Paul Stoffregen.

Basic code for encoder:

[Arduino_sketches/03_Keyboard_Encoder]

Tasks:

1. Print a different keyboard character depending upon which way the Encoder rotates
2. Connect up push button on Encoder shaft

More information on this library:

- https://www.pjrc.com/teensy/td_libs_Encoder.html

Include Encoder library

Create Encoder

Read current value

If value changed, print to Keyboard

```
#include <Keyboard.h>
#include <Encoder.h>

// Create Encoder
Encoder myEnc(2,3);

void setup(){

   // ⚠ Safety stop!
   // Program will not begin unless pin 12 connected to GND
   pinMode(12, INPUT_PULLUP);
   while(digitalRead(12)==HIGH){
      // do nothing!
      delay(500);
   }

   // Start keyboard
   Keyboard.begin();
}

long last_position = 0;

void loop() {
   long current_position = myEnc.read();

   if (current_position != last_position){
      Keyboard.println(current_position);
      last_position = current_position;
   }
}
```